HR Data Analysis

By Mohamed Jamyl

http://linkedin.com/in/mohamed-jamyl

https://www.kaggle.com/mohamedjamyl

https://github.com/Mohamed-Jamyl

In [1]:
```
from IPython.display import Image
Image(filename='hr.JPG')
```

Out[1]:

Import Libraries

In [2]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Exploratory Data Analysis (EDA)

---

1 - Initial Data Understanding

- **Data loading and Inspection**
- **Data Types**
- **Missing Values**
- **Duplicates**

In [3]:
```python
import pandas as pd
df = pd.read_csv('HR Data.csv')
```

In [4]:
```python
df
```

Out[4]:

	EmployeeID	EmployeeName	Salary	Position	State	DateOfBirth	Gender	MaritalStatus	HiringDate	TerminationDate	Emp
0	1	John Smith	62506	Production Technician I	MA	7/10/1983	M	Single	7/5/2011	NaN	
1	2	Sarah Johnson	104437	Sr. DBA	MA	5/5/1975	M	Married	3/30/2015	6/16/2016	
2	3	Michael Williams	64955	Production Technician II	MA	9/19/1988	F	Married	7/5/2011	9/24/2012	
3	4	Emily Brown	64991	Production Technician I	MA	9/27/1988	F	Married	1/7/2008	NaN	
4	5	David Jones	50825	Production Technician I	MA	9/8/1989	F	Divorced	7/11/2011	9/6/2016	
...	...	...	...	...	...	...	...	...	...	...	
306	307	Nana Asare	65893	Production Technician II	MA	5/11/1985	M	Single	7/7/2014	NaN	
307	308	Yaa Yeboah	48513	Production Technician I	MA	5/4/1982	F	Single	9/2/2008	9/29/2015	
308	309	Kojo Ofori	220450	CIO	MA	8/30/1979	F	Single	4/10/2010	NaN	
309	310	Esi Amoako	89292	Data Analyst	MA	2/24/1979	F	Single	3/30/2015	NaN	
310	311	Kweku Annan	45046	Production Technician I	MA	8/17/1978	F	Widowed	9/29/2014	NaN	

311 rows × 16 columns

In [5]:
```python
df.shape
```

Out[5]:
```
(311, 16)
```

In [6]:
```python
df.columns
```

Out[6]:
```
Index(['EmployeeID', 'EmployeeName', 'Salary', 'Position', 'State',
       'DateOfBirth', 'Gender', 'MaritalStatus', 'HiringDate',
       'TerminationDate', 'EmploymentStatus', 'Department',
       'RecruitmentSource', 'PerformanceScore', 'EngagementSurvey',
       'EmployeeSatisfaction'],
      dtype='object')
```

In [7]:
```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 311 entries, 0 to 310
Data columns (total 16 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   EmployeeID            311 non-null    int64  
 1   EmployeeName          311 non-null    object 
 2   Salary                311 non-null    int64  
 3   Position              311 non-null    object 
 4   State                 311 non-null    object 
 5   DateOfBirth           311 non-null    object 
 6   Gender                311 non-null    object 
 7   MaritalStatus         311 non-null    object 
 8   HiringDate            311 non-null    object 
 9   TerminationDate       104 non-null    object 
 10  EmploymentStatus      311 non-null    object 
 11  Department            311 non-null    object 
 12  RecruitmentSource     311 non-null    object 
 13  PerformanceScore      311 non-null    object 
 14  EngagementSurvey      311 non-null    float64
 15  EmployeeSatisfaction  311 non-null    int64  
dtypes: float64(1), int64(3), object(12)
memory usage: 39.0+ KB
```

In [8]:
```python
df.isnull().sum()
```

Out[8]:
```
EmployeeID                0
EmployeeName              0
Salary                    0
Position                  0
State                     0
DateOfBirth               0
Gender                    0
MaritalStatus             0
HiringDate                0
TerminationDate         207
EmploymentStatus          0
Department                0
RecruitmentSource         0
PerformanceScore          0
EngagementSurvey          0
EmployeeSatisfaction      0
dtype: int64
```

In [9]:
```python
df.duplicated().sum()
```

Out[9]:
```
np.int64(0)
```

---

2 - Basic Statistical Overview

- Summary Statistical : **Describe()**

In [10]:
```python
df.describe().T
```

Out[10]:

	count	mean	std	min	25%	50%	75%	max
EmployeeID	311.0	156.000000	89.922189	1.00	78.50	156.00	233.5	311.0
Salary	311.0	69020.684887	25156.636930	45046.00	55501.50	62810.00	72036.0	250000.0
EngagementSurvey	311.0	4.110000	0.789938	1.12	3.69	4.28	4.7	5.0
EmployeeSatisfaction	311.0	3.890675	0.909241	1.00	3.00	4.00	5.0	5.0

In [11]:
```python
df.select_dtypes(include='object').describe().T
```

Out[11]:

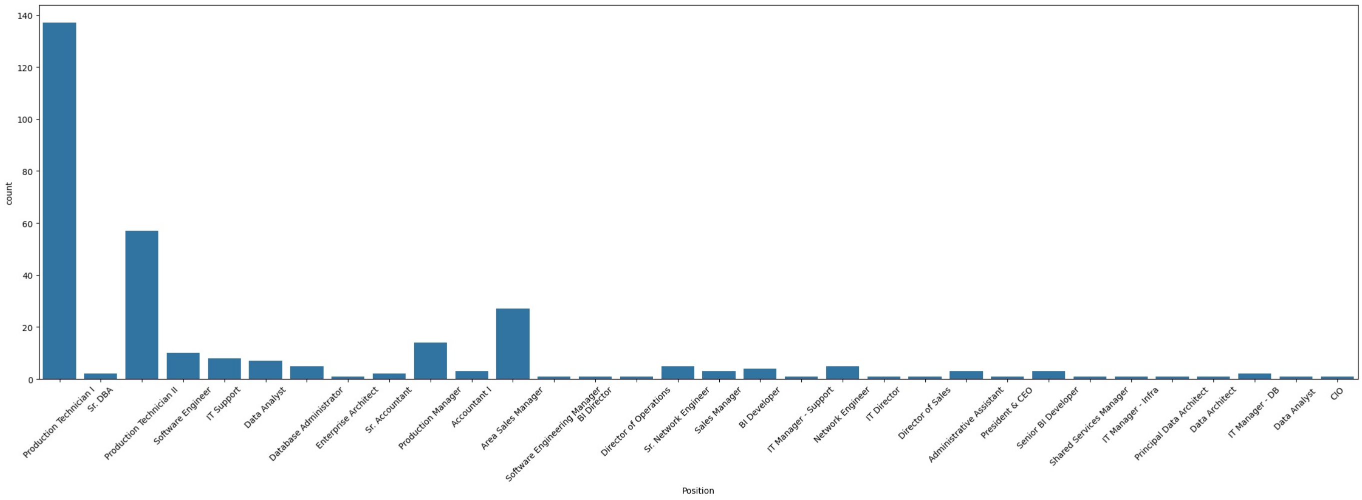
	count	unique	top	freq
EmployeeName	311	310	Christopher Wilson	2
Position	311	32	Production Technician I	137
State	311	28	MA	276
DateOfBirth	311	307	9/22/1976	2
Gender	311	3	F	176
MaritalStatus	311	5	Single	137
HiringDate	311	101	1/10/2011	14
TerminationDate	104	96	9/24/2012	2
EmploymentStatus	311	3	Active	207
Department	311	6	Production	209
RecruitmentSource	311	9	Indeed	87
PerformanceScore	311	4	Fully Meets	243

- Summary Statistical : **Value_counts()**

In [12]:
```python
df['Position'].value_counts()
```

Out[12]:
```
Position
Production Technician I        137
Production Technician II        57
Area Sales Manager              27
Production Manager              14
Software Engineer               10
IT Support                       8
Data Analyst                     7
Database Administrator           5
Sr. Network Engineer             5
Network Engineer                 5
BI Developer                     4
Accountant I                     3
Administrative Assistant         3
Sales Manager                    3
Senior BI Developer              3
Sr. Accountant                   2
Sr. DBA                          2
IT Manager - DB                  2
Software Engineering Manager     1
Enterprise Architect             1
Director of Operations           1
BI Director                      1
IT Manager - Support             1
IT Director                      1
President & CEO                  1
Director of Sales                1
IT Manager - Infra               1
Shared Services Manager          1
Principal Data Architect         1
Data Architect                   1
Data Analyst                     1
CIO                              1
Name: count, dtype: int64
```

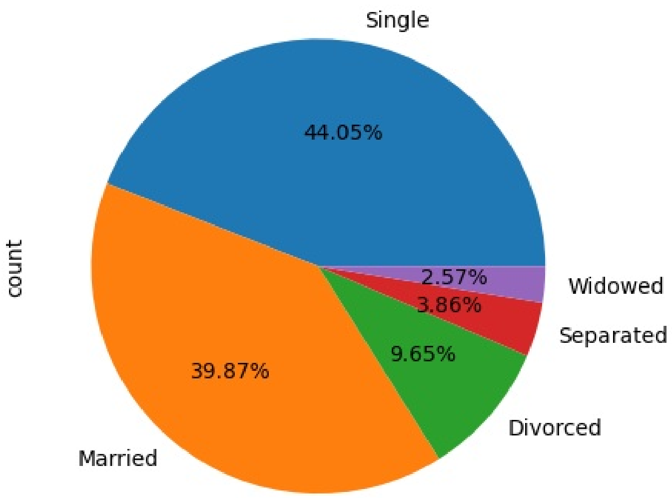
In [13]:
```python
plt.figure(figsize=(28,8))
sns.countplot(data = df, x = 'Position')
plt.xticks(rotation=45)
plt.show()
```



In [14]:
```python
df['MaritalStatus'].value_counts()
```

Out[14]:
```
MaritalStatus
Single       137
Married      124
Divorced      30
Separated     12
Widowed        8
Name: count, dtype: int64
```

In [15]:
```python
df['MaritalStatus'].value_counts().plot.pie(autopct='%0.2f%%')
plt.show()
```

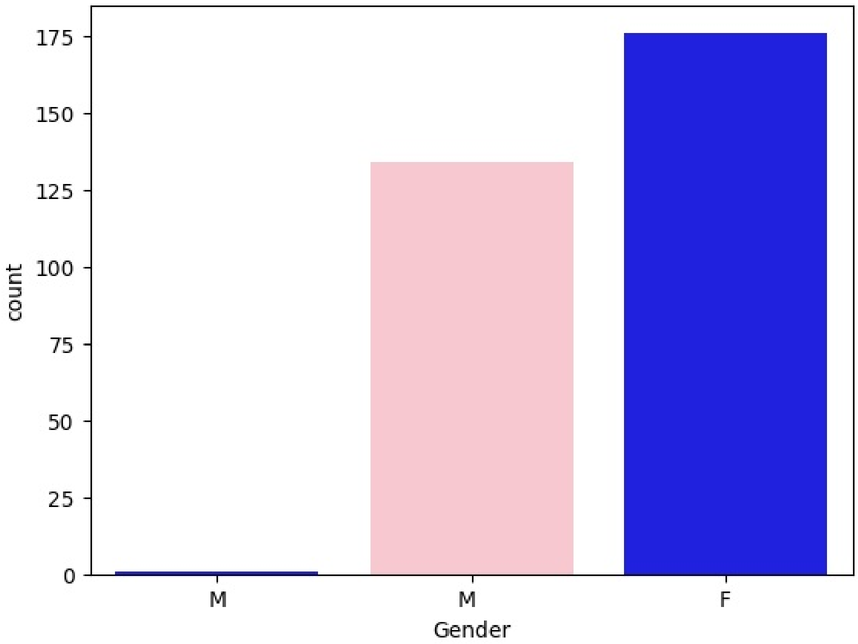


In [16]:
```
df['Gender'].value_counts()
```

Out[16]:
```
Gender
F     176
M     134
M       1
Name: count, dtype: int64
```

In [17]:
```
sns.countplot(data=df, x='Gender', palette=['blue','pink'])
plt.show()
```

```
C:\Users\RPC\AppData\Local\Temp\ipykernel_19172\1138742225.py:1: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable
to `hue` and set `legend=False` for the same effect.

  sns.countplot(data=df, x='Gender', palette=['blue','pink'])
C:\Users\RPC\AppData\Local\Temp\ipykernel_19172\1138742225.py:1: UserWarning:
The palette list has fewer values (2) than needed (3) and will cycle, which may produce an uninterpretable plot.
  sns.countplot(data=df, x='Gender', palette=['blue','pink'])
```



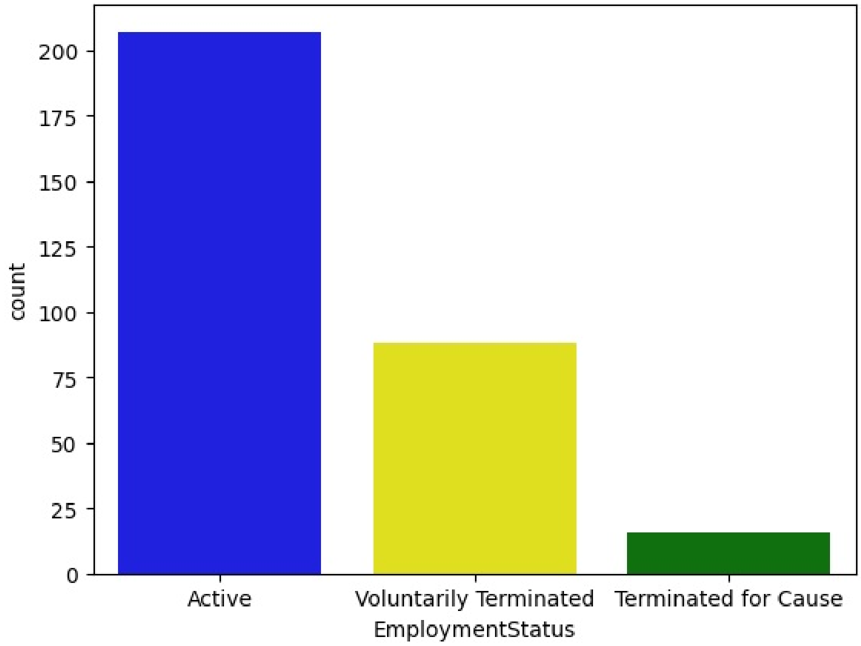
In [18]:
```
df.EmploymentStatus.value_counts()
```

Out[18]:
```
EmploymentStatus
Active                    207
Voluntarily Terminated     88
Terminated for Cause       16
Name: count, dtype: int64
```

In [19]:
```
sns.countplot(data=df, x='EmploymentStatus',palette=['blue','yellow','green'])
```

```
plt.show()
```

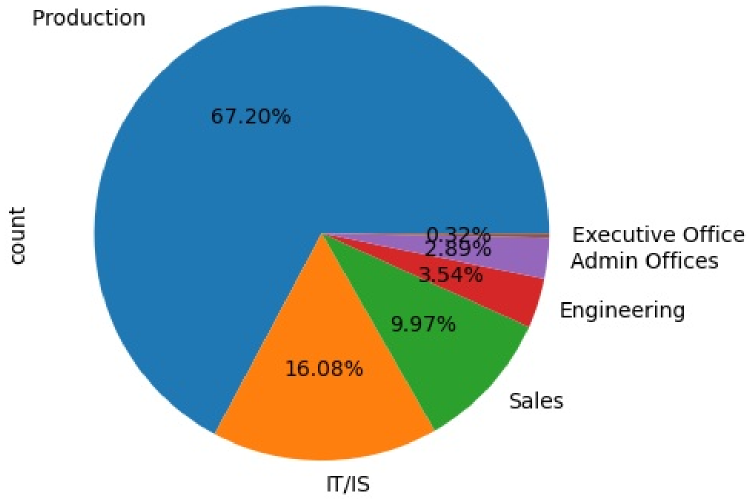
```
C:\Users\RPC\AppData\Local\Temp\ipykernel_19172\679451588.py:1: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.countplot(data=df, x='EmploymentStatus',palette=['blue','yellow','green'])
```



In [20]:
```
df.Department.value_counts()
```

Out[20]:
```
Department
Production          209
IT/IS                50
Sales                31
Engineering          11
Admin Offices         9
Executive Office      1
Name: count, dtype: int64
```

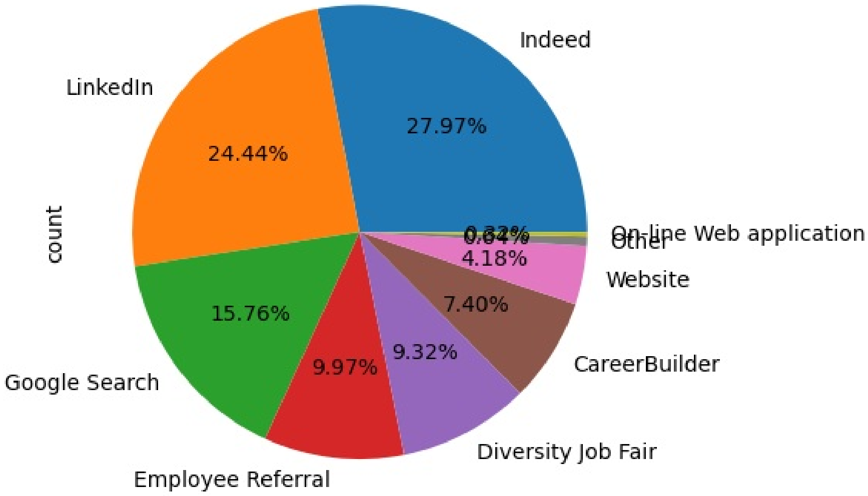
In [21]:
```
df.Department.value_counts().plot.pie(autopct='%0.2f%%')
plt.show()
```



In [22]:
```
df.RecruitmentSource.value_counts()
```

Out[22]:
```
RecruitmentSource
Indeed                     87
LinkedIn                   76
Google Search              49
Employee Referral          31
Diversity Job Fair         29
CareerBuilder              23
Website                    13
Other                       2
On-line Web application     1
Name: count, dtype: int64
```

In [23]:
```
df.RecruitmentSource.value_counts().plot.pie(autopct='%0.2f%%')
plt.show()
```

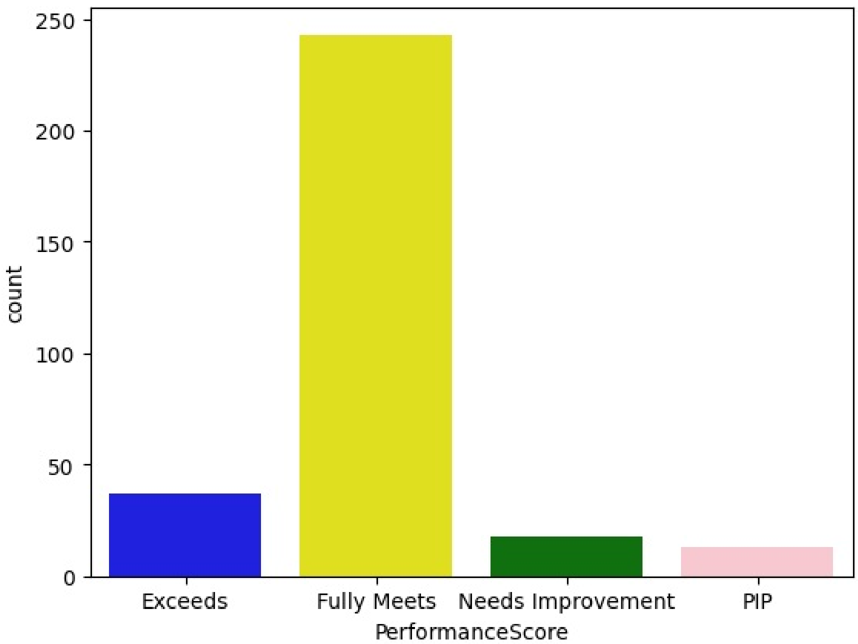


In [24]:
```
df.PerformanceScore.value_counts()
```

Out[24]:
```
PerformanceScore
Fully Meets          243
Exceeds               37
Needs Improvement     18
PIP                   13
Name: count, dtype: int64
```

In [25]:
```
sns.countplot(data=df, x='PerformanceScore', palette=['blue','yellow','green','pink'])
plt.show()
```

```
C:\Users\RPC\AppData\Local\Temp\ipykernel_19172\1171278047.py:1: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable
to `hue` and set `legend=False` for the same effect.

  sns.countplot(data=df, x='PerformanceScore', palette=['blue','yellow','green','pink'])
```

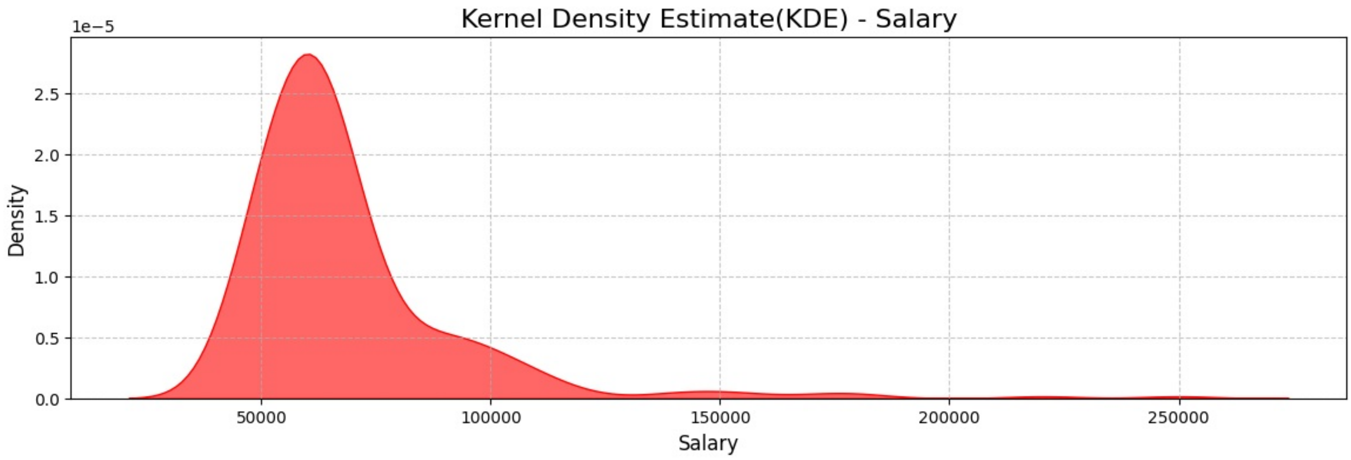


---

3 - Distribution of Variables

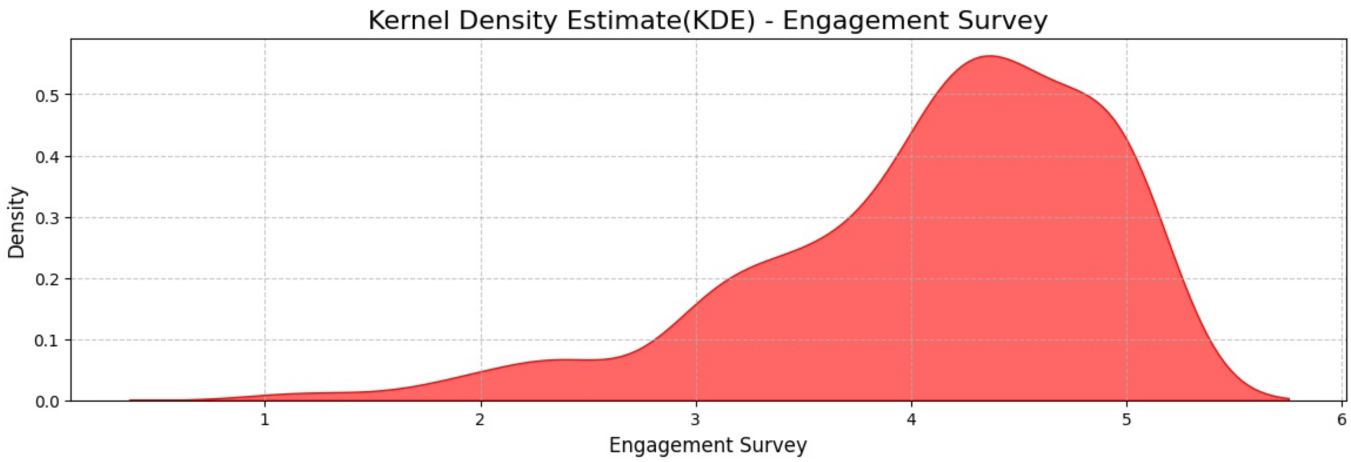
- **Numerical Features (KDE)**

In [26]:
```
plt.figure(figsize=(14,4))
sns.kdeplot(df['Salary'], fill = True, color='red',alpha=0.6)
plt.title('Kernel Density Estimate(KDE) - Salary', fontsize= 16)
plt.xlabel('Salary', fontsize= 12)
plt.ylabel('Density', fontsize=12)
plt.grid(True, linestyle = '--', alpha = 0.7)
plt.show()
```



- **Shape**: The distribution is heavily right-skewed. There's a prominent peak (mode) around $60,000, indicating that a large proportion of the salaries fall around this value.
- **Peak/Mode**: The highest density (around 2.8×10 −5) occurs at approximately $60,000.
- **Spread/Range**: Salaries range from slightly below $50, 000 *to over* 250,000, though the density of higher salaries is very low.
- **Tail**: There's a long, thin tail extending to the right, suggesting that while most salaries are concentrated at the lower end, there are a few individuals earning significantly higher salaries. This is typical for salary distributions, where a small number of high earners can pull the average up.
- **Interpretation**: The plot suggests that the majority of individuals in this dataset earn a salary in the range of roughly $50, 000 *to* 100,000, with a strong concentration around $60, 000. *There are progressively fewer individuals as salary increases beyond* 100,000.

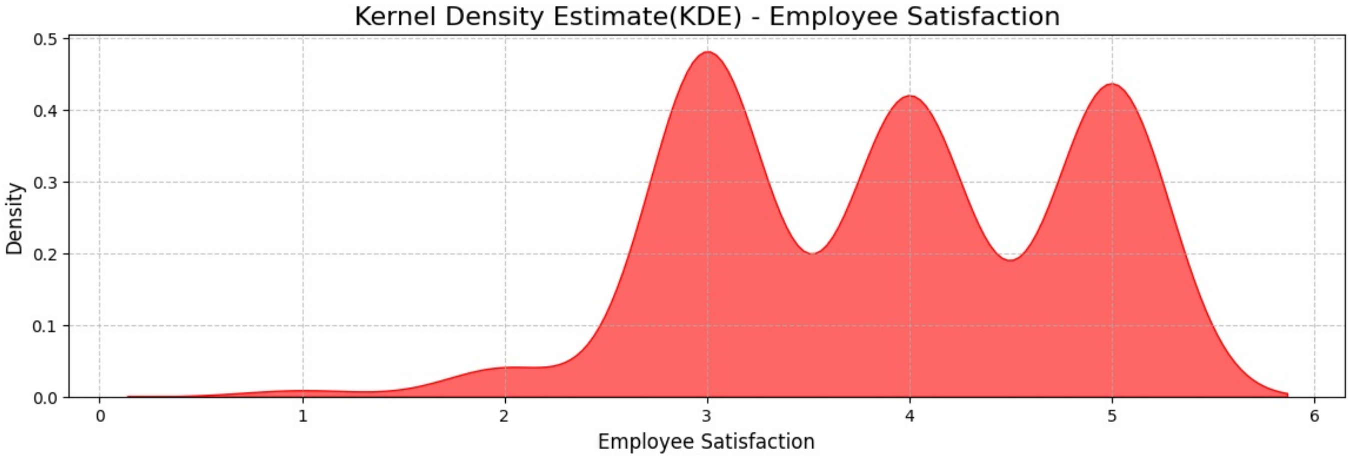
In [27]:
```
plt.figure(figsize=(14,4))
sns.kdeplot(df['EngagementSurvey'], fill = True, color='red',alpha=0.6)
plt.title('Kernel Density Estimate(KDE) - Engagement Survey', fontsize= 16)
plt.xlabel('Engagement Survey', fontsize= 12)
plt.ylabel('Density', fontsize=12)
plt.grid(True, linestyle = '--', alpha = 0.7)
plt.show()
```



- **Shape:** The distribution is left-skewed, meaning there's a longer tail on the left side and a concentration of data points on the right side.
- **Peak/Mode:** The highest density (around 0.55) occurs at approximately an "Engagement Survey" score of 4.5. This indicates that a large number of survey responses are clustered around this higher engagement score.
- **Range:** The engagement survey scores range from slightly below 0.5 to a maximum of 6.
- **Concentration:** There's a significant concentration of scores between approximately 3.5 and 5.5, with the peak around 4.5. This suggests that most respondents reported relatively high engagement.
- **Tail:** There's a gradual decrease in density as the scores go lower than 3.5, indicating fewer respondents with very low engagement scores.
- **Interpretation:** The plot suggests that the overall engagement level is quite high, with the majority of individuals giving scores towards the upper end of the scale. While there are some lower scores, they are much less frequent.

In [28]:
```
plt.figure(figsize=(14,4))
sns.kdeplot(df['EmployeeSatisfaction'], fill = True, color='red',alpha=0.6)
plt.title('Kernel Density Estimate(KDE) - Employee Satisfaction', fontsize= 16)
plt.xlabel('Employee Satisfaction', fontsize= 12)
```

```
plt.ylabel('Density', fontsize=12)
plt.grid(True, linestyle = '--', alpha = 0.7)
plt.show()
```

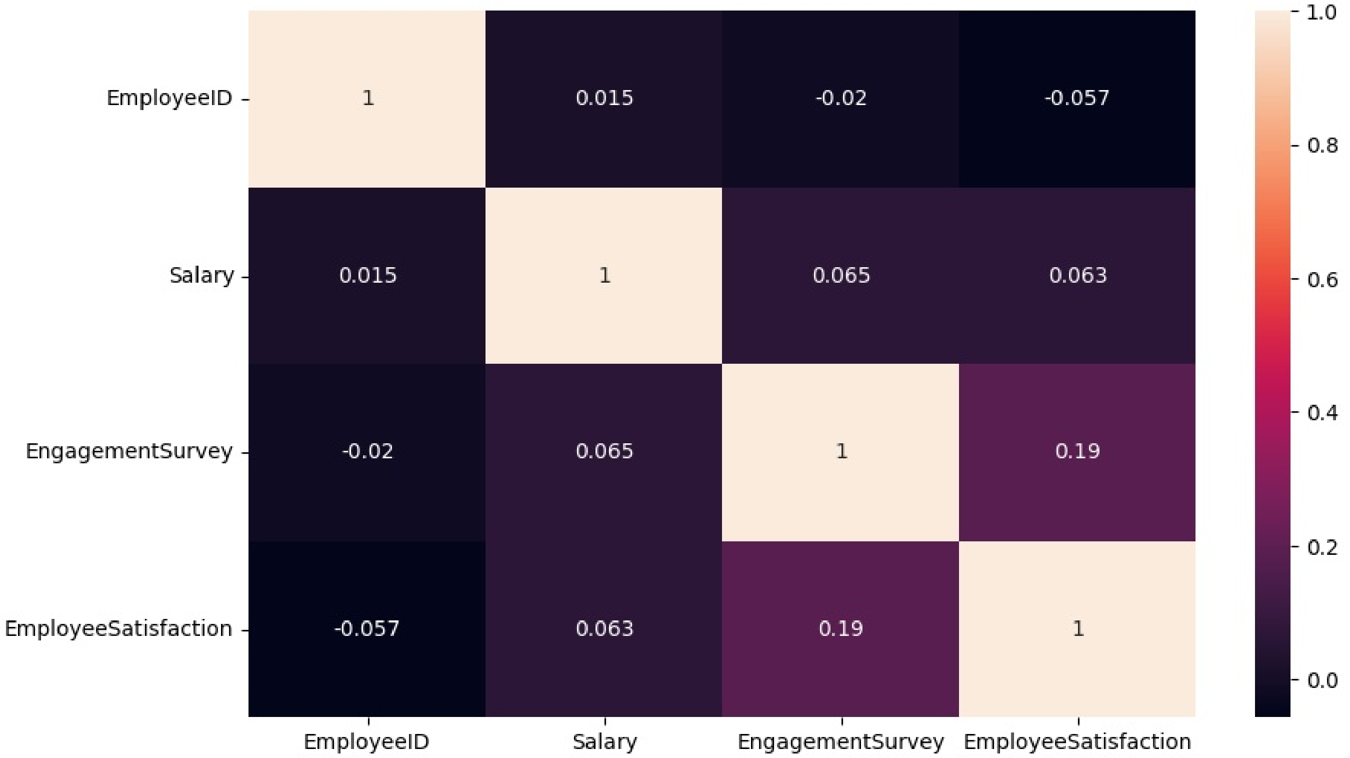


- **Shape:** The distribution is multimodal, specifically tri-modal, meaning it has three distinct peaks. This suggests that employee satisfaction scores tend to cluster around three different levels.
- **Peaks/Modes:**
- The first and most prominent peak (highest density, around 0.48) is located at an "Employee Satisfaction" score of approximately 3.0.
- The second peak (density around 0.42) is around 4.0.
- The third peak (density around 0.44) is around 5.0.
- **Range:** Employee satisfaction scores range from approximately 0 to 6.
- **Interpretation:** The multimodal nature of this distribution is quite interesting. It suggests that, rather than a single general level of satisfaction, employees tend to fall into three main groups:
- A significant group of employees with a satisfaction score of around 3.0 (perhaps indicating moderate satisfaction).
- Another substantial group with a satisfaction score of around 4.0 (indicating good satisfaction).
- A third, also substantial, group with a satisfaction score of around 5.0 (indicating very high satisfaction).

This type of distribution could imply different segments within the employee population with varying levels of satisfaction, or perhaps different factors influencing satisfaction that lead to these distinct clusters. It would be valuable to investigate why these three distinct groups exist.

Ckecking Correlation between the features

In [29]:
```
plt.figure(figsize = (10,6))
sns.heatmap(df.select_dtypes(include='number').corr(), annot=True)
plt.show()
```



- Most of the correlations between these variables are very weak, close to zero.
- The strongest (though still weak) positive correlation is observed between EngagementSurvey and EmployeeSatisfaction (0.19). This indicates that there's some positive relationship between how engaged employees feel and their overall satisfaction, but it's not a strong one.
- EmployeeID shows no meaningful correlation with any other variable, which is expected.
- Salary has almost no linear relationship with EngagementSurvey or EmployeeSatisfaction in this dataset.

Data Cleaning

In [30]:
```
df['Gender'].value_counts()
```

Out[30]:
```
Gender
F      176
M      134
M        1
Name: count, dtype: int64
```

In [31]:
```
def replaceGender(gender):
    gender = str(gender)

    if gender == 'M':
        return 'Male'

    if gender == 'M ':
        return 'Male'

    if gender == 'F':
        return 'Female'

df['gender'] = df['Gender'].apply(lambda x : replaceGender(x))
```

In [32]:
```
df['gender']
```

Out[32]:
```
0        Male
1        Male
2      Female
3      Female
4      Female
        ...
306      Male
307    Female
308    Female
309    Female
310    Female
Name: gender, Length: 311, dtype: object
```

In [33]:
```
df['gender'].value_counts()
```

Out[33]:
```
gender
Female    176
Male      135
Name: count, dtype: int64
```

In [34]:
```
df
```

Out[34]:

	EmployeeID	EmployeeName	Salary	Position	State	DateOfBirth	Gender	MaritalStatus	HiringDate	TerminationDate	Emp
0	1	John Smith	62506	Production Technician I	MA	7/10/1983	M	Single	7/5/2011	NaN	
1	2	Sarah Johnson	104437	Sr. DBA	MA	5/5/1975	M	Married	3/30/2015	6/16/2016	
2	3	Michael Williams	64955	Production Technician II	MA	9/19/1988	F	Married	7/5/2011	9/24/2012	
3	4	Emily Brown	64991	Production Technician I	MA	9/27/1988	F	Married	1/7/2008	NaN	
4	5	David Jones	50825	Production Technician I	MA	9/8/1989	F	Divorced	7/11/2011	9/6/2016	
...	...	...	...	...	...	...	...	...	...	...	
306	307	Nana Asare	65893	Production Technician II	MA	5/11/1985	M	Single	7/7/2014	NaN	
307	308	Yaa Yeboah	48513	Production Technician I	MA	5/4/1982	F	Single	9/2/2008	9/29/2015	
308	309	Kojo Ofori	220450	CIO	MA	8/30/1979	F	Single	4/10/2010	NaN	
309	310	Esi Amoako	89292	Data Analyst	MA	2/24/1979	F	Single	3/30/2015	NaN	
310	311	Kweku Annan	45046	Production Technician I	MA	8/17/1978	F	Widowed	9/29/2014	NaN	

311 rows × 17 columns

In [35]:
```
df[df['EmployeeName'] == 'Christopher Smith']
```

Out[35]:

	EmployeeID	EmployeeName	Salary	Position	State	DateOfBirth	Gender	MaritalStatus	HiringDate	TerminationDate	Emplo
150	151	Christopher Smith	250000	President & CEO	MA	9/21/1954	F	Married	7/2/2012	NaN	

In [36]:
```
# change gender from female to male
df.loc[df['EmployeeName'] == 'Christopher Smith', 'gender'] = 'Male'
```

In [37]:
```
df[df['EmployeeName'] == 'Christopher Smith']
```

Out[37]:

	EmployeeID	EmployeeName	Salary	Position	State	DateOfBirth	Gender	MaritalStatus	HiringDate	TerminationDate	Emplo
150	151	Christopher Smith	250000	President & CEO	MA	9/21/1954	F	Married	7/2/2012	NaN	

In [38]:
```
df = df.drop(['TerminationDate','DateOfBirth','Gender','EmployeeID'], axis = 1)
```

Extracting features

In [39]:
```
df['EmployeeSatisfaction'].value_counts()
```

Out[39]:
```
EmployeeSatisfaction
3    108
5     98
4     94
2      9
1      2
Name: count, dtype: int64
```

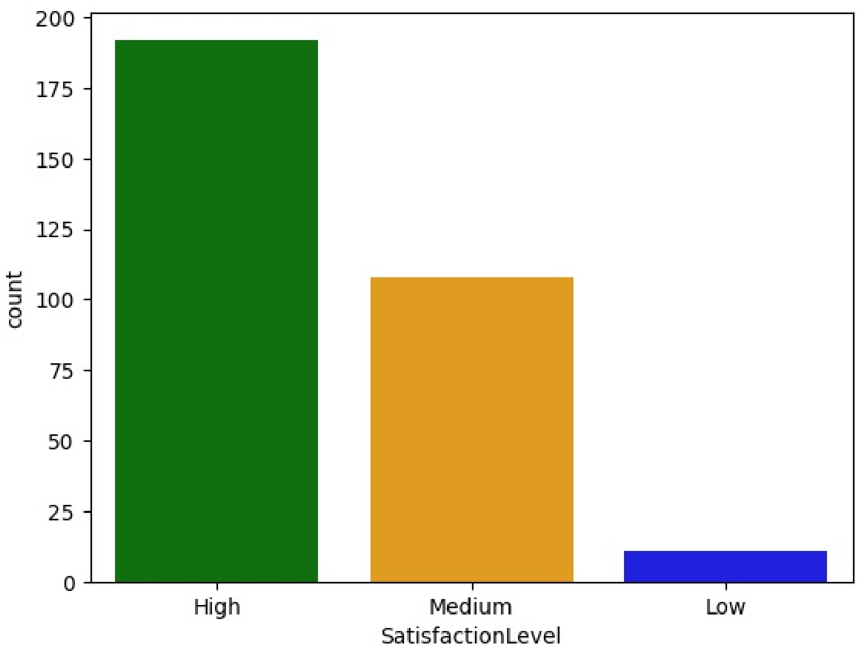
In [40]:
```
def SatisfactionLevel(i):
    i = int(i)
    if i <= 2:
        return 'Low'
    elif i == 3 :
        return 'Medium'
    else:
        return 'High'

df['SatisfactionLevel'] = df['EmployeeSatisfaction'].apply(lambda x : SatisfactionLevel(x))
```

In [41]:
```
df['SatisfactionLevel'].value_counts()
```

Out[41]:
```
SatisfactionLevel
High      192
Medium    108
Low        11
Name: count, dtype: int64
```

In [42]:
```
sns.countplot(data=df, x= 'SatisfactionLevel', palette=['green','orange','blue'])
plt.show()
```

```
C:\Users\RPC\AppData\Local\Temp\ipykernel_19172\1911257579.py:1: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable
to `hue` and set `legend=False` for the same effect.

  sns.countplot(data=df, x= 'SatisfactionLevel', palette=['green','orange','blue'])
```



- **Extract Length of Service Tenure**

In [43]:
```
df['HiringDate'][0].split('/')[2]
```

Out[43]: '2011'

In [44]:
```
df['HiringYear'] = df['HiringDate'].apply(lambda x: x.split('/')[2])
```

In [45]:
```
df['HiringYear'] = df['HiringYear'].astype(int)
```

In [46]:
```
df['Length of Service Tenure'] = df['HiringYear'].apply(lambda x : 2025 - x)
```

In [47]:
```
df[['EmployeeName','Length of Service Tenure']]
```

Out[47]:

	EmployeeName	Length of Service Tenure
0	John Smith	14
1	Sarah Johnson	10
2	Michael Williams	14
3	Emily Brown	17
4	David Jones	14
...	...	...
306	Nana Asare	11
307	Yaa Yeboah	17
308	Kojo Ofori	15
309	Esi Amoako	10
310	Kweku Annan	11

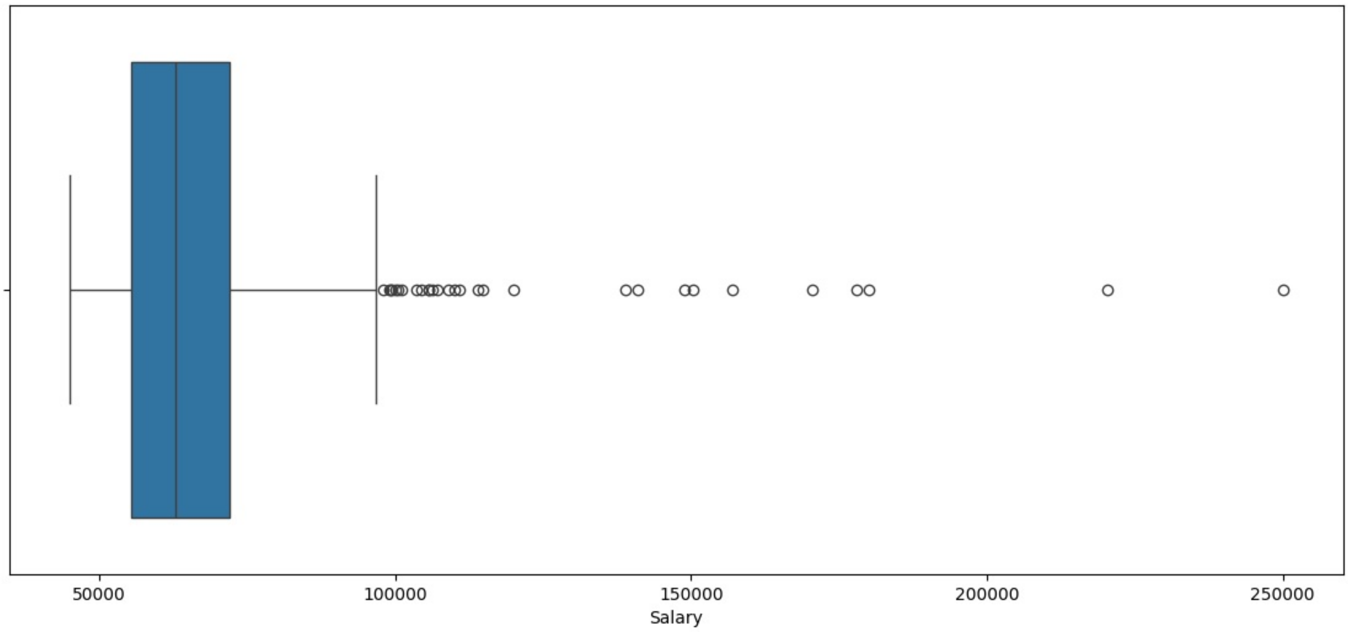
311 rows × 2 columns

Detect Outliers

In [48]:
```
df['Salary'].describe()
```

Out[48]:
```
count       311.000000
mean      69020.684887
std       25156.636930
min       45046.000000
25%       55501.500000
50%       62810.000000
75%       72036.000000
max      250000.000000
Name: Salary, dtype: float64
```

In [49]:
```
plt.figure(figsize=(14,6))
sns.boxplot(data=df, x = 'Salary')
plt.show()
```

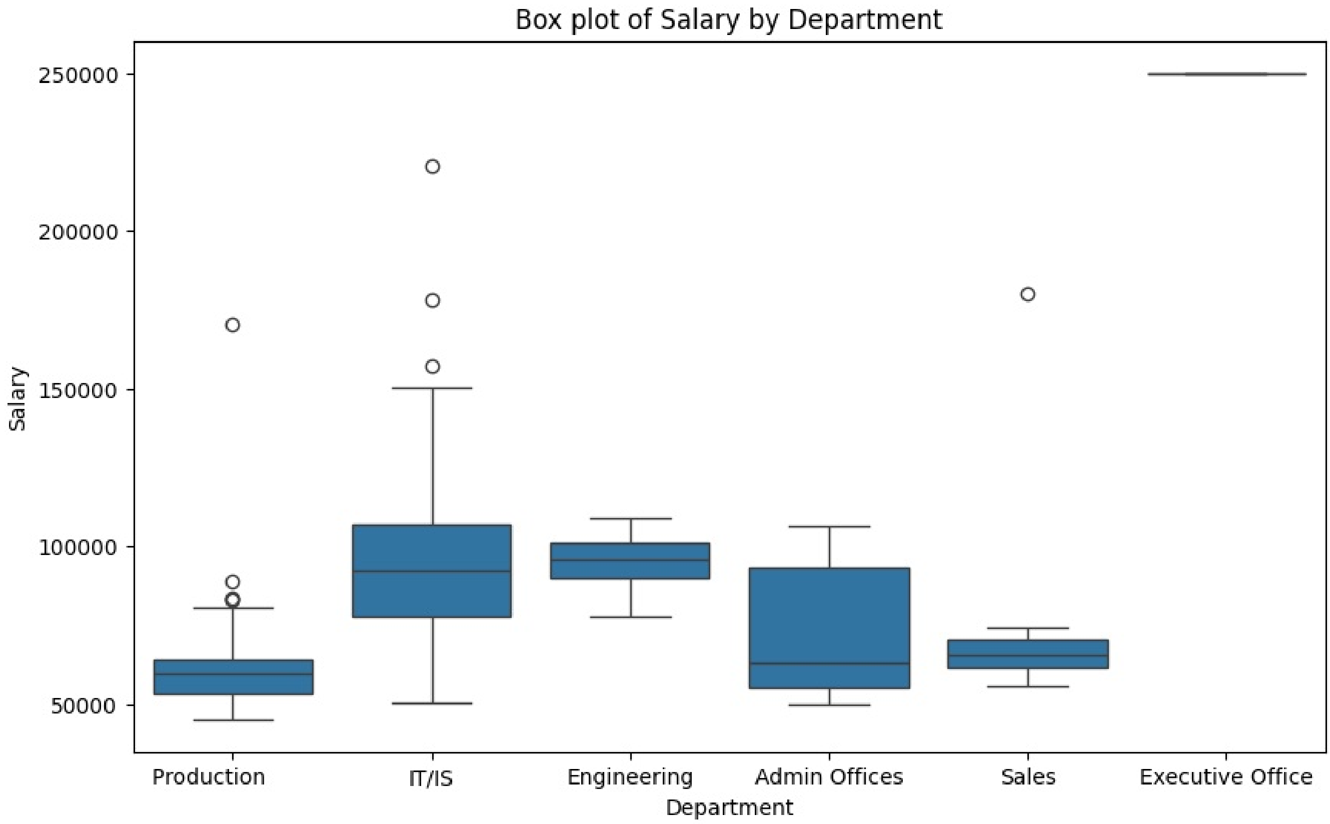


- **Median:** The line inside the box represents the median salary. It appears to be around $60,000.
    - Box (Interquartile Range - IQR): The box itself spans from the first quartile (Q1) to the third quartile (Q3).
    - Q1 is approximately $50,000.
    - Q3 is approximately $75,000.
    - The box is relatively small, indicating that 50% of the salaries are concentrated within a narrow range.
- **Whiskers:** The whiskers extend to the minimum and maximum values that are not considered outliers.

- The right whisker extends up to roughly $100,000. This indicates the vast majority of salaries are below this value.
- The left whisker extends to a value slightly above $40,000.

- **Outliers:** The circles to the right of the right whisker represent outliers. These are salary values that are significantly higher than the rest of the data. They extend far to the right, with some values reaching up to approximately $250,000.

In [50]:
```
fig, axes = plt.subplots(nrows=1,ncols=1)
fig.set_size_inches(10,6)
sns.boxplot(data=df, y='Salary', x ='Department', orient='v', ax=axes)
axes.set(xlabel = 'Department' ,ylabel= 'Salary',title = 'Box plot of Salary by Department')
plt.show()
```



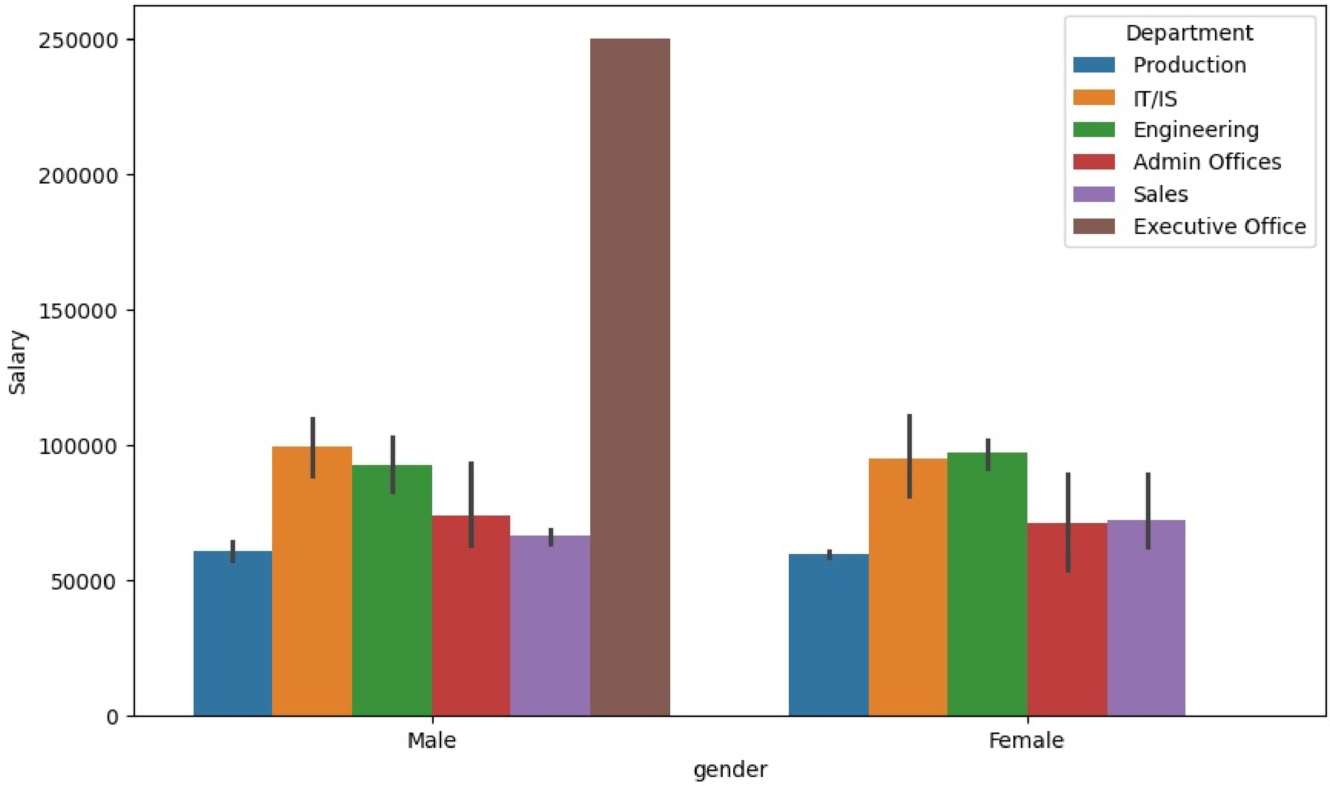
- **Production:**
  - Median salary is relatively low, around $60,000.
  - The interquartile range (IQR, the height of the box) is small, indicating a tight clustering of salaries.
  - There are some high-salary outliers, with one reaching approximately $170,000.
- **IT/IS:**
  - The median salary is higher than Production, around $80,000.
  - This department has a wider salary range and a larger IQR compared to Production, suggesting more variability in salaries.
  - There are several high-salary outliers, with some going up to nearly $220,000.
- **Engineering:**
  - The median salary is one of the highest, close to $95,000.
  - The IQR is relatively small, showing that salaries are tightly clustered around the median.
  - No outliers are visible.
- **Admin Offices:**
  - The median salary is around $65,000.
  - The salary range is wide, with a large IQR.
  - The highest salary is slightly above $100,000, and there are no visible outliers.
- **Sales:**
  - The median salary is around $65,000, similar to Admin Offices.

- The IQR is the smallest among all departments, indicating a very narrow range of salaries for the middle 50% of employees.
- There is one outlier with a salary around $180,000.

- **Executive Office:**
  - This department has a single line rather than a full box plot, which indicates that all salaries are clustered at a single value.
  - The salary is at the very top of the chart, approximately $250,000. This is an expected finding, as executive roles typically command the highest salaries and there are often very few employees in this department.

Analysis

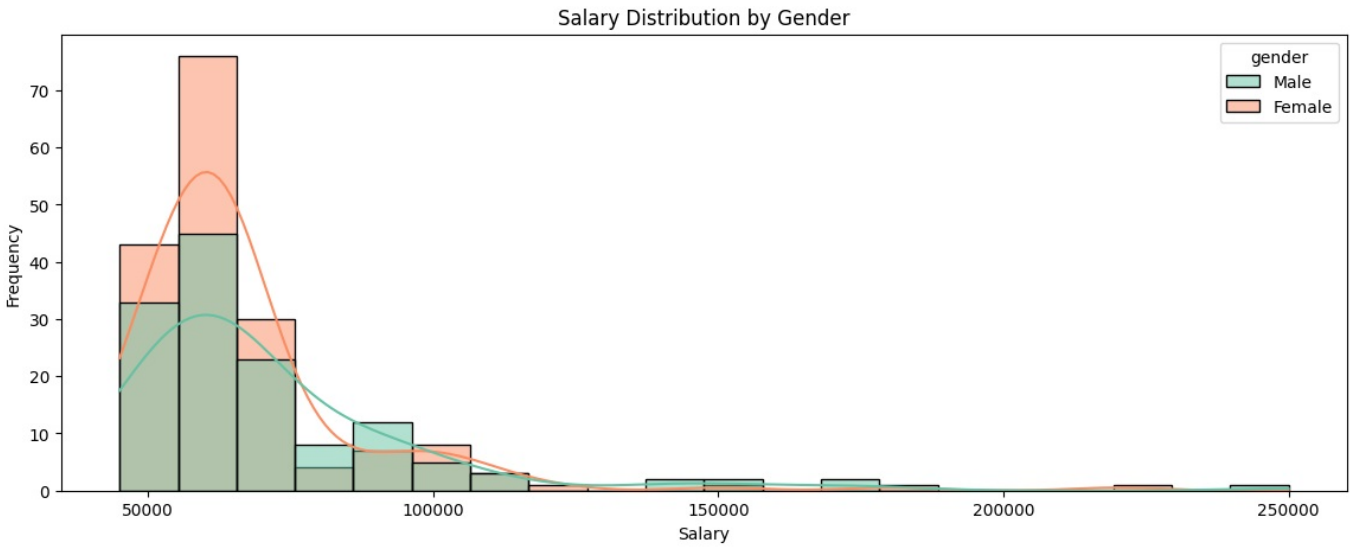
In [51]:
```
# price dependency on
plt.figure(figsize=(10,6))
sns.barplot(data = df, x= 'gender', y='Salary',hue='Department')
plt.show()
```



- **Gender Comparison:**
  - For most departments, the average salary is very similar between males and females.
  - The "IT/IS" and "Engineering" departments show slightly higher average salaries for females, although the difference is minor.
  - The "Production," "Admin Offices," and "Sales" departments have almost identical average salaries for both genders.
  - There is no data for the "Executive Office" for females, which is consistent with the earlier box plot analysis suggesting a very small number of individuals in that department, possibly all male in this dataset.
- **Department Comparison (for each gender):**
- **Male:**
  - The "Executive Office" has a drastically higher average salary than any other department, which is an expected finding for this type of role.
  - "IT/IS" and "Engineering" have the next highest average salaries, followed by "Admin Offices" and "Sales."
  - "Production" has the lowest average salary for males.
- **Female:**
  - The "Engineering" and "IT/IS" departments have the highest average salaries for females.
  - "Admin Offices," "Sales," and "Production" have the lowest average salaries for females, with very similar means.
- **Standard Deviation (Error Bars):**

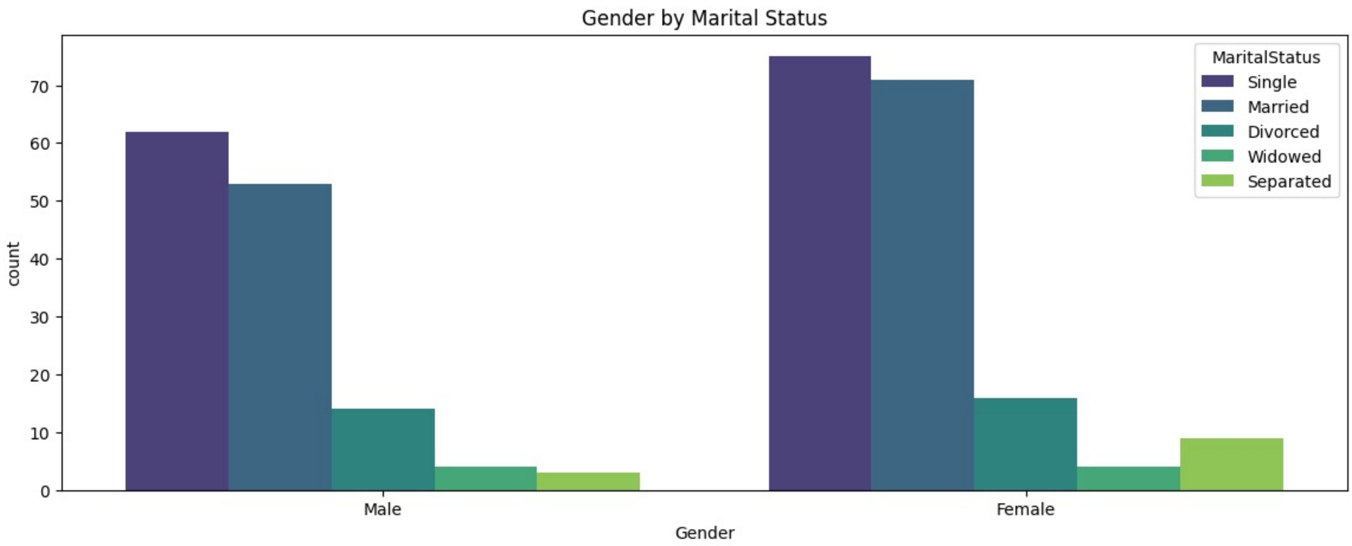
- The "IT/IS" and "Engineering" departments have larger error bars for both genders compared to other departments, indicating a wider spread or greater variability in salaries within these departments. This is particularly noticeable for males in the "IT/IS" department.
- The "Production," "Admin Offices," and "Sales" departments have relatively small error bars, suggesting less variability in salaries.
- The "Executive Office" has no visible error bar for males, confirming the earlier box plot observation that all salaries are clustered at a single value.

In [52]:
```
plt.figure(figsize=(14, 5))
sns.histplot(data=df, x='Salary', hue='gender', kde=True, bins=20, palette='Set2')
plt.title('Salary Distribution by Gender')
plt.xlabel('Salary')
plt.ylabel('Frequency')
plt.show()
```



- **Overall Distribution:** Both male and female salary distributions are heavily right-skewed, with the majority of individuals earning lower salaries and a long tail extending to the right for higher salaries. This is consistent with the earlier analysis of the overall salary distribution.
- **Male Salary Distribution (Green/Teal):**
  - The highest frequency for males is in the salary range of approximately 60, 000*to*70,000.
  - The KDE curve shows a peak around this range.
  - There's a gradual decrease in frequency as salaries increase, with a few individuals in the higher salary brackets (e.g., above $150,000).
- **Female Salary Distribution (Orange/Salmon):**
  - The highest frequency for females is in a slightly lower salary range, around 50, 000*to*60,000.
  - The KDE curve for females also shows a peak in this range.
  - Similar to males, the frequency decreases as salaries increase, but the distribution seems to have a slightly lower concentration at the peak compared to males.
- **Comparison between Genders:**
  - The most significant difference is in the mode (the peak of the distribution). The most common salary for males is slightly higher than the most common salary for females.
  - The distribution of males appears to be more concentrated at its peak, while the female distribution is slightly flatter at its peak.
  - Both genders have individuals in the very high-salary brackets (outliers), and the overall shape of the distributions is very similar. The plot does not show a dramatic difference in salary range or distribution shape between genders, but it does indicate that the central tendency (the most frequent salary) for males is slightly higher than for females. This is consistent with the grouped bar chart's findings where male salaries were slightly higher on average in some departments.

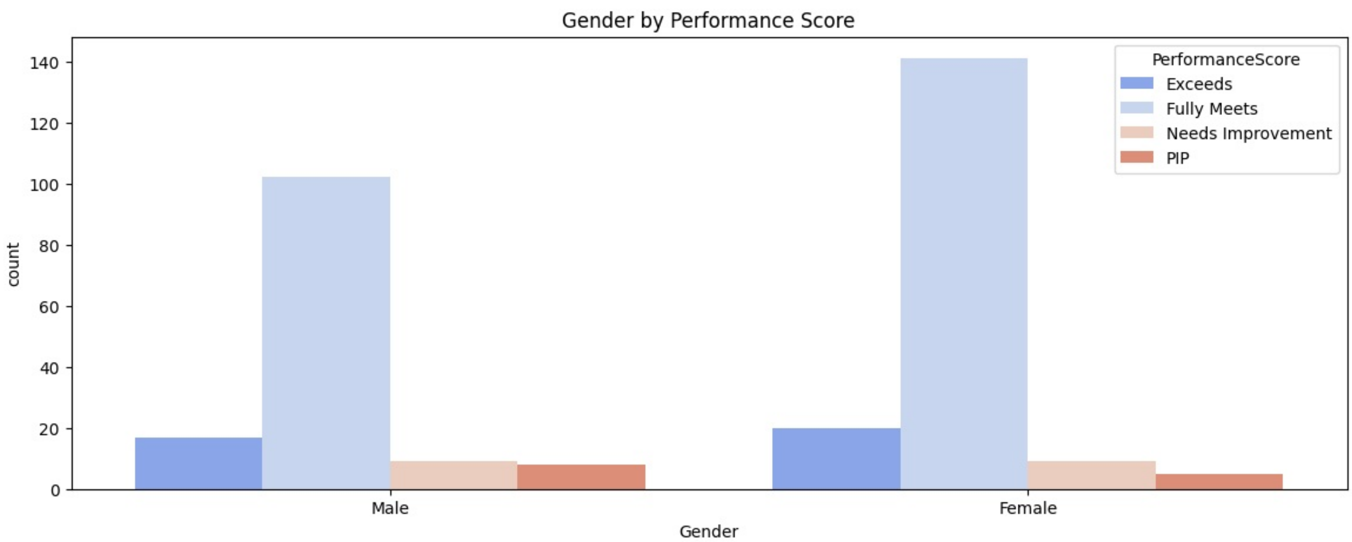
In [53]:
```
plt.figure(figsize=(14,5))
sns.countplot(data = df, x = 'gender', hue='MaritalStatus',palette='viridis')
plt.title('Gender by Marital Status')
plt.xlabel('Gender')
plt.show()
```



- **Gender Distribution:** The plot shows a roughly equal count of male and female employees in the dataset, with a slightly higher number of females overall.
- **Breakdown by Marital Status:**
  - **Single**: The largest group for both genders is "Single". There are more single females than single males.
  - **Married:** "Married" is the second largest group for both genders. The number of married females is slightly higher than the number of married males.
  - **Divorced:** "Divorced" is the third largest group. There are more divorced females than divorced males.
  - **Widowed:** "Widowed" is a small group, with more females than males.
  - **Separated:** "Separated" is a very small group. The count of separated females is similar to that of widowed females, while the count of separated males is slightly lower.

In [54]:

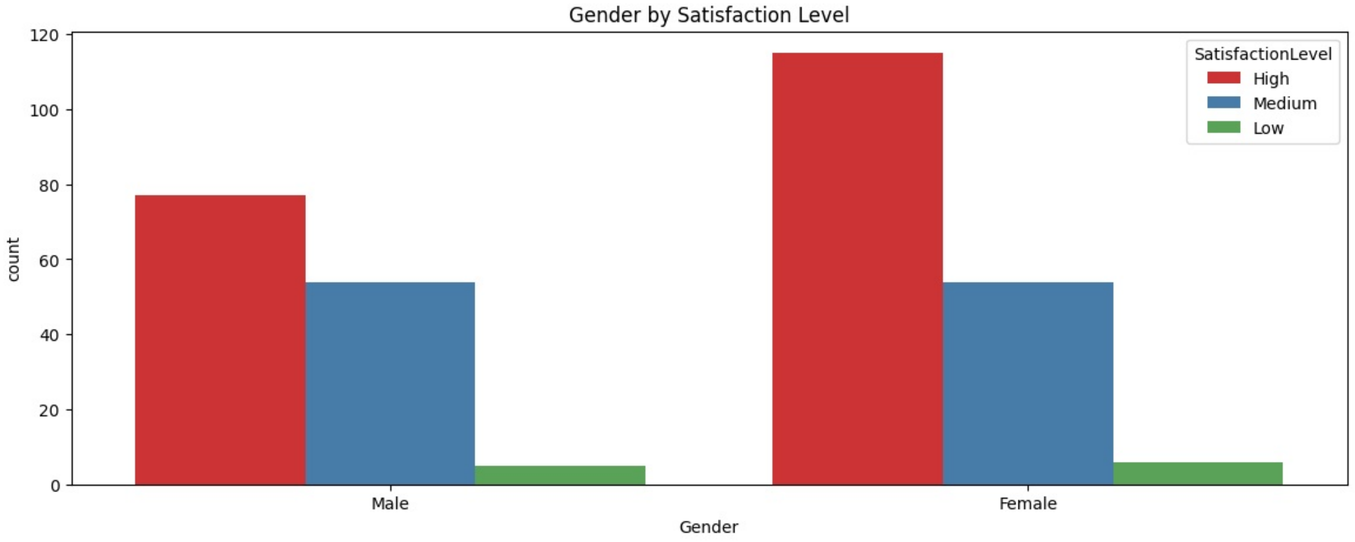
```
plt.figure(figsize=(14,5))
sns.countplot(data = df, x = 'gender', hue='PerformanceScore',palette='coolwarm')
plt.title('Gender by Performance Score')
plt.xlabel('Gender')
plt.show()
```



- **Overall Performance:** The majority of employees, both male and female, have a "Fully Meets" performance score. This is the tallest bar for each gender group.
- **Gender Comparison:**
  - **Fully Meets:** The number of females who "Fully Meet" expectations (over 140) is significantly higher than the number of males (around 100) who do.
  - **Exceeds:** A slightly higher number of females (around 20) "Exceed" expectations compared to males (around 17).
  - **Needs Improvement:** The number of males and females in the "Needs Improvement" category is very similar, though males are slightly more numerous.
  - **PIP (Performance Improvement Plan):** The number of males and females on a "PIP" is also very similar, with a slightly higher count for males.

- **Breakdown within each Gender:**
  - **Male:** The distribution is heavily skewed towards "Fully Meets," followed by "Exceeds," and then a small number of employees in the "Needs Improvement" and "PIP" categories.
  - **Female:** The distribution for females is similar, but the count for "Fully Meets" is much higher than for any other score. The number of females who "Exceed" is also higher than the number who "Needs Improvement" or are on a "PIP".

In [55]:
```
plt.figure(figsize=(14,5))
sns.countplot(data = df, x = 'gender', hue='SatisfactionLevel',palette='Set1')
plt.title('Gender by Satisfaction Level')
plt.xlabel('Gender')
plt.show()
```



- **Overall Satisfaction:** For both male and female employees, the "High" satisfaction level has the highest count, followed by "Medium," and then a very small number of employees with a "Low" satisfaction level. This suggests that the overall employee satisfaction in the company is generally positive.
- **Gender Comparison:**
  - **High Satisfaction:** There are significantly more females (around 115) who have a "High" satisfaction level compared to males (around 75).
  - **Medium Satisfaction:** The number of females with "Medium" satisfaction (around 55) is very similar to the number of males with "Medium" satisfaction (around 55).
  - **Low Satisfaction:** The count of females with "Low" satisfaction (around 5) is slightly higher than the count of males with "Low" satisfaction (around 4).
- **Breakdown within each Gender:**
  - **Male:** The distribution shows a strong majority of males with "High" satisfaction, followed by "Medium," with "Low" satisfaction being a very small minority.
  - **Female:** The trend is even more pronounced for females, with a large majority having "High" satisfaction, followed by "Medium," and a very small number having "Low" satisfaction.

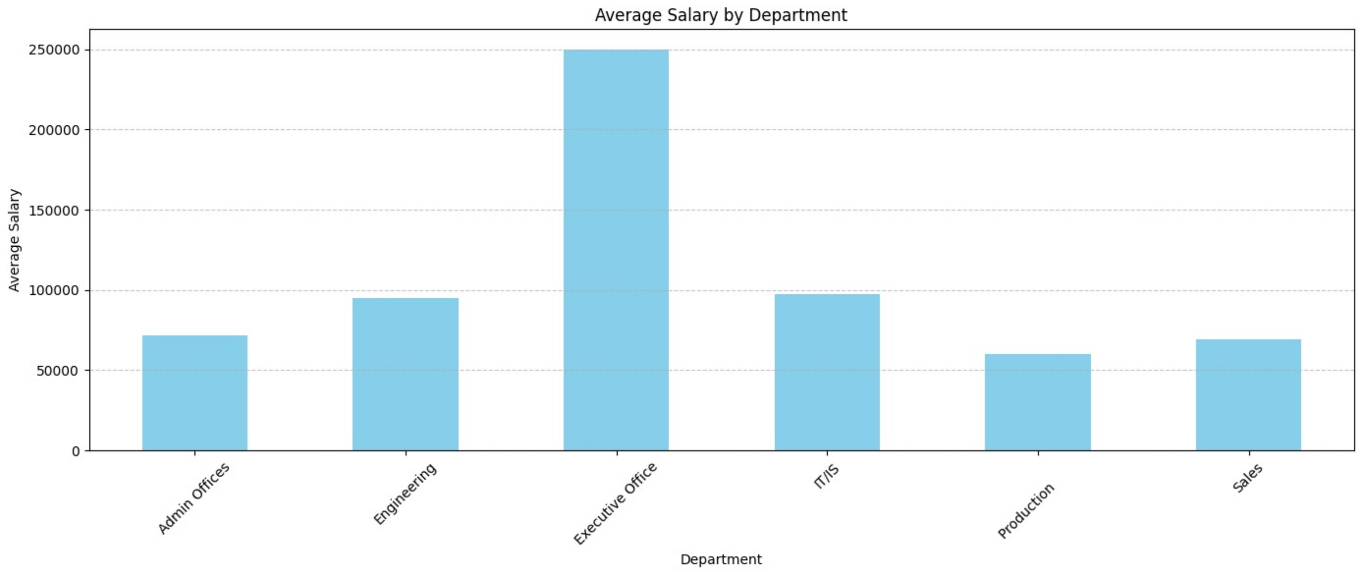
Mean of salary Group by department

In [57]:
```
DepartmentBySalary = df.groupby(by='Department')['Salary'].mean()
DepartmentBySalary
```

Out[57]:
```
Department
Admin Offices         71791.888889
Engineering           94989.454545
Executive Office     250000.000000
IT/IS                 97064.640000
Production            59953.545455
Sales                 69061.258065
Name: Salary, dtype: float64
```

In [60]:
```
DepartmentBySalary.plot(kind='bar', figsize=(14, 6), color='skyblue')
plt.title('Average Salary by Department')
plt.xlabel('Department')
plt.ylabel('Average Salary')
plt.xticks(rotation=45)
plt.grid(axis='y', linestyle='--', alpha=0.7)
```

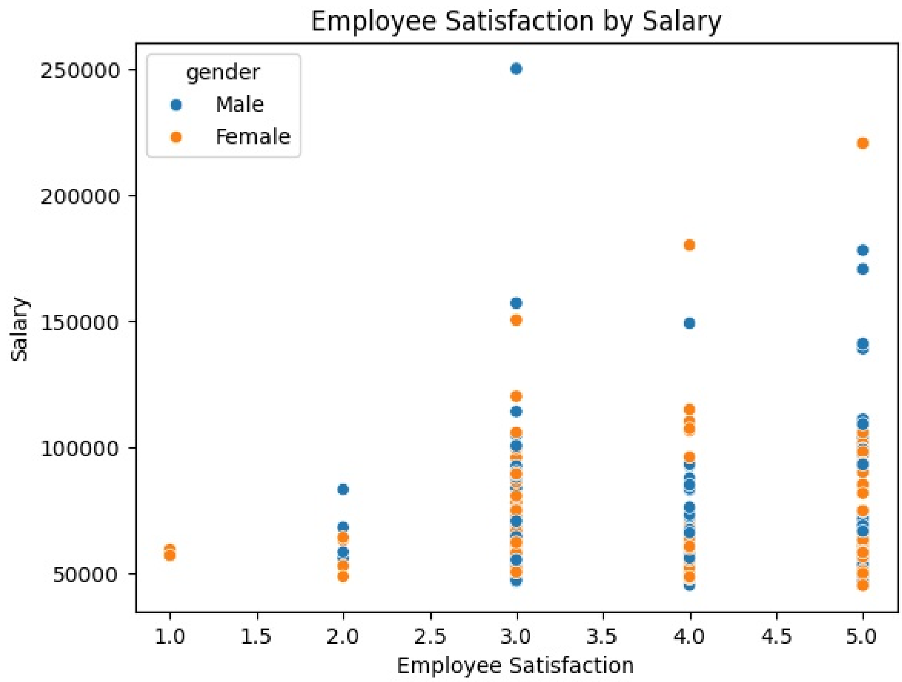
```
plt.tight_layout()
plt.show()
```



- **Overall:** The overall salary distribution is heavily right-skewed, with a prominent peak around $60,000. This indicates that most employees earn a salary in this range.
- **By Department:** There are significant differences in average salary across departments.
  - The Executive Office has a disproportionately high average salary, around $250,000, which is expected for senior leadership roles.
  - The Engineering and IT/IS departments have the next highest average salaries, at around 95, 000*and*98,000 respectively.
  - The Admin Offices, Production, and Sales departments have similar, lower average salaries, all falling below $75,000.

Eamplyee Satisfaction by Salary

In [63]:
```
sns.scatterplot(data=df, x='EmployeeSatisfaction',y='Salary',hue='gender')
plt.title('Employee Satisfaction by Salary')
plt.xlabel('Employee Satisfaction')
plt.ylabel('Salary')
plt.show()
```



- **Male (Blue Dots):**
  - Male employees are represented across all five satisfaction levels.
  - At the highest satisfaction level (5), male employees show a wide range of salaries, including some of the highest salaries in the dataset (e.g., above 150, 000*andevenreachingcloseto*200,000).

- There is a single male employee with a satisfaction rating of 2 and a salary of around $85,000.
- A significant number of male employees fall into the middle salary range (approximately $50, 000 to$125,000) at satisfaction levels 3, 4, and 5.

- **Female (Orange Dots):**
  - Female employees are also represented across all five satisfaction levels.
  - At the highest satisfaction level (5), female employees also have a wide salary range. Notably, there is a female employee with one of the highest salaries in the dataset, exceeding $200,000.
  - At satisfaction level 3, there is a female employee with a salary close to $150,000, which is higher than many other employees at this satisfaction level.
  - There is a data point for a female employee with the lowest satisfaction score (1), who has a salary of approximately $58,000.
  - Similar to males, many female employees are clustered in the mid-to-high salary range at satisfaction levels 3, 4, and 5.

Filtering data to Satisfaction status greater than or equal 4

In [68]:
```
SatisfactionScoreGTe4 = df[df['EmployeeSatisfaction']>=4]
```

In [69]:
```
SatisfactionScoreGTe4
```

Out[69]:

	EmployeeName	Salary	Position	State	MaritalStatus	HiringDate	EmploymentStatus	Department	RecruitmentSource	Perf
0	John Smith	62506	Production Technician I	MA	Single	7/5/2011	Active	Production	LinkedIn	
3	Emily Brown	64991	Production Technician I	MA	Married	1/7/2008	Active	Production	Indeed	
4	David Jones	50825	Production Technician I	MA	Divorced	7/11/2011	Voluntarily Terminated	Production	Google Search	
5	Jessica Davis	57568	Production Technician I	MA	Single	1/9/2012	Active	Production	LinkedIn	
7	Ashley Wilson	59365	Production Technician I	MA	Widowed	9/30/2013	Active	Production	Employee Referral	
...	...	...	...	...	...	...	...	...	...	
303	Ama Asante	59728	Production Technician I	MA	Single	1/9/2012	Voluntarily Terminated	Production	Diversity Job Fair	
305	Abena Yeboah	60446	Production Technician II	MA	Single	9/29/2014	Active	Production	LinkedIn	
306	Nana Asare	65893	Production Technician II	MA	Single	7/7/2014	Active	Production	LinkedIn	
308	Kojo Ofori	220450	CIO	MA	Single	4/10/2010	Active	IT/IS	Employee Referral	
310	Kweku Annan	45046	Production Technician I	MA	Widowed	9/29/2014	Active	Production	LinkedIn	

192 rows × 16 columns

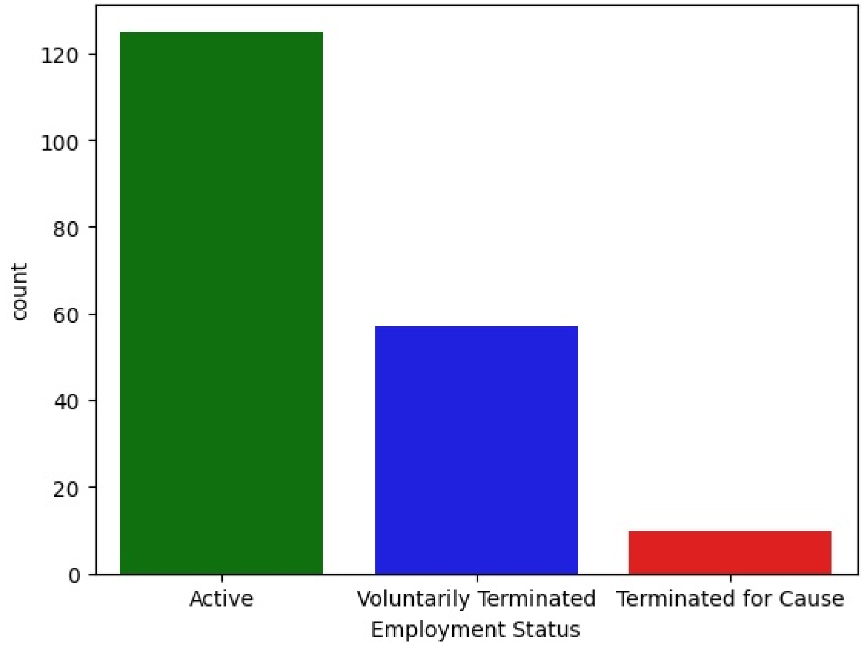
In [70]:
```
SatisfactionScoreGTe4.shape
```

Out[70]:
```
(192, 16)
```

- The HR dataset reveals that most employees are **highly satisfied** and perform at a level that **"fully meets expectations"**.

In [74]:
```
sns.countplot(data = SatisfactionScoreGTe4, x = 'EmploymentStatus', palette=['green','blue','red'])
plt.xlabel('Employment Status')
plt.show()
```

```
C:\Users\RPC\AppData\Local\Temp\ipykernel_19172\4235329608.py:1: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable
to `hue` and set `legend=False` for the same effect.

  sns.countplot(data = SatisfactionScoreGTe4, x = 'EmploymentStatus', palette=['green','blue','red'])
```



Filtering data to Satisfaction status less than or equal 2

In [75]:
```
SatisfactionScoreLTe2 = df[df['EmployeeSatisfaction']<=2]
```

In [76]:
```
SatisfactionScoreLTe2
```

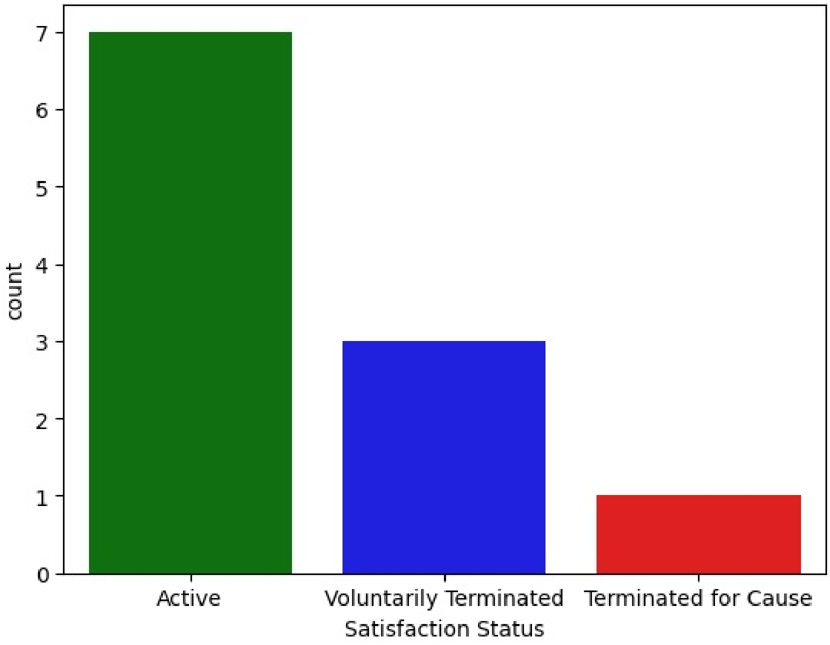
Out[76]:

	EmployeeName	Salary	Position	State	MaritalStatus	HiringDate	EmploymentStatus	Department	RecruitmentSource	Perf
29	Michelle Moore	63000	Accountant I	MA	Married	10/27/2008	Active	Admin Offices	Diversity Job Fair	
54	Kevin Scott	68051	Production Manager	MA	Divorced	7/20/2010	Active	Production	CareerBuilder	
69	Kayla Coleman	53189	Production Technician I	MA	Married	7/7/2014	Active	Production	Indeed	
72	Patrick Lopez	59231	Area Sales Manager	WA	Single	2/20/2012	Active	Sales	Website	
83	Courtney Adams	56847	Production Technician II	MA	Separated	7/7/2014	Active	Production	Indeed	
137	Melissa Bennett	83082	Production Manager	MA	Married	2/21/2011	Voluntarily Terminated	Production	Indeed	
188	Penelope Evans	55800	Production Technician II	MA	Single	8/15/2011	Voluntarily Terminated	Production	LinkedIn	
205	Jack Hill	52674	Production Technician I	MA	Single	3/31/2014	Terminated for Cause	Production	LinkedIn	
263	Kwabena Boateng	64021	Production Technician I	MA	Married	2/20/2012	Active	Production	Indeed	
267	Kweku Asante	58273	Area Sales Manager	NV	Married	5/12/2014	Active	Sales	Website	
307	Yaa Yeboah	48513	Production Technician I	MA	Single	9/2/2008	Voluntarily Terminated	Production	Google Search	

In [77]:
```
sns.countplot(data = SatisfactionScoreLTe2, x= 'EmploymentStatus', palette=['green','blue','red'])
plt.xlabel('Satisfaction Status')
```

```
plt.show()
```

```
C:\Users\RPC\AppData\Local\Temp\ipykernel_19172\3940468111.py:1: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable
to `hue` and set `legend=False` for the same effect.

  sns.countplot(data = SatisfactionScoreLTe2, x= 'EmploymentStatus', palette=['green','blue','red'])
```



Conclusions and Recommendations

- The HR dataset reveals that most employees are **highly satisfied** and perform at a level that **"fully meets expectations"**.
- Gender differences in salary are minor, with **Engineering** and **IT/IS** departments offering the highest salaries regardless of gender.
- Performance and satisfaction are weakly correlated, suggesting other unmeasured factors (like leadership, work-life balance) might influence performance.
- ### **Recommendation:** Management could use such predictive models to:
  - Identify employees at risk of poor performance.
  - Design personalized training or incentive plans.
  - Improve employee retention strategies.

In [ ]: